

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平11-306793 ✓

(43)公開日 平成11年(1999)11月5日

(51)Int.Cl.⁸

G 1 1 C 29/00

G 0 6 T 7/00

識別記号

6 5 5

F I

G 1 1 C 29/00

G 0 6 F 15/62

6 5 5 D

4 0 5 A

審査請求 未請求 請求項の数11 O L (全 11 頁)

(21)出願番号

特願平10-116627

(22)出願日

平成10年(1998)4月27日

(71)出願人 390005175

株式会社アドバンテスト

東京都練馬区旭町1丁目32番1号

(72)発明者 藤原 輝明

東京都練馬区旭町1丁目32番1号 株式会
社アドバンテスト内

(72)発明者 丸尾 和幸

東京都練馬区旭町1丁目32番1号 株式会
社アドバンテスト内

(72)発明者 山口 隆弘

東京都練馬区旭町1丁目32番1号 株式会
社アドバンテスト内

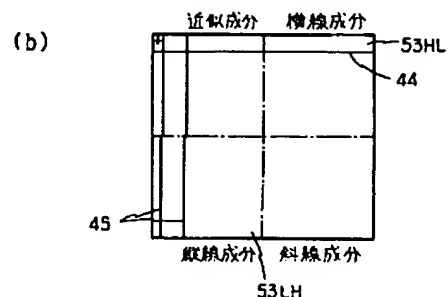
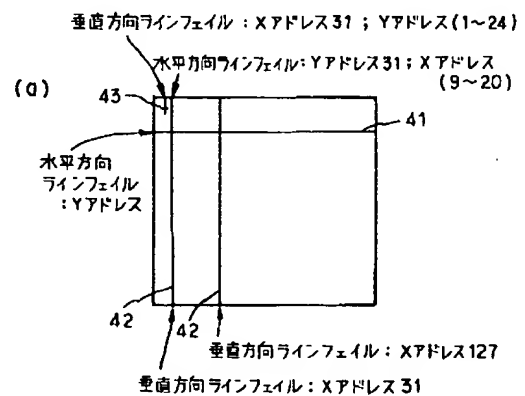
(74)代理人 弁理士 草野 卓 (外1名)

(54)【発明の名称】 不良解析方法および装置

(57)【要約】

【課題】 ICメモリ試験結果の不良ビットマップから不良アドレス、不良個数を自動的に求める。

【解決手段】 不良ビットマップを二次元ウェーブレット変換し(S0)、そのX方向ハイパス、Y方向ローパス情報(縦方向成分)X_{HL}(i, j)を、各iについてY方向に積算してヒストグラムを作り(S2)、その積算値≠0のアドレスiから不良アドレスXを求め(S4)、その積算値の絶対値から不良数を求めてそれぞれ出力し(S5)、更に各不良アドレスについて不良ビットマップをY方向に走査して良から不良へ変った、また、その逆に変った各Yアドレスと、その間の不良数を出力する。



【特許請求の範囲】

【請求項1】 フェイルビットマップから不良情報を抽出する不良解析装置であって、
上記入力フェイルビットマップを二次元ウェーブレット変換する二次元ウェーブレット変換手段と、
上記ウェーブレット変換結果を画像情報として出力する画像出力装置と、
を具備することを特徴とする不良解析装置。

【請求項2】 フェイルビットマップから不良情報を抽出する不良解析装置であって、
上記入力フェイルビットマップを二次元ウェーブレット変換する二次元ウェーブレット変換手段と、
上記二次元ウェーブレット変換結果中のX方向ローパス・Y方向ハイパス情報についてY方向にウェーブレット係数を積算してXアドレスヒストグラムを生成し、X方向ハイパス・Y方向ローパス情報についてX、Y方向にウェーブレット係数を積算してYアドレスヒストグラムを生成する、ヒストグラム生成手段と、
を具備することを特徴とする不良解析装置。

【請求項3】 請求項2記載の不良解析装置において、ヒストグラム生成手段にて得られたXアドレスヒストグラム中の積算値が0でないXアドレスを、Yアドレスヒストグラム中の積算値が0でないYアドレスをそれぞれ不良のアドレスとして、そのXアドレスのヒストグラム積算値、Yアドレスのヒストグラム積算値をそれぞれ不良の個数として出力する不良情報出力手段と、
を具備することを特徴とする不良解析装置。

【請求項4】 請求項3の不良解析装置において、不良の個数が、入力フェイルビットマップのXアドレスまたはYアドレスの大きさに等しいか否かを判定する手段と、
その判定手段が等しいと判定すると不良はフェイルビットマップの対応XアドレスまたはYアドレス全体に渡っているものであると出力する手段とを具備することを特徴とする不良解析装置。

【請求項5】 請求項3の不良解析装置において、上記判定手段が等しくない判定すると、入力フェイルビットマップ、あるいはフェイルビットマップをウェーブレット変換した結果の中の、不良が存在するXアドレスまたはYアドレスを走査し、不良が始まるアドレスと、その不良が終わるアドレスとを検出する不良アドレス検出手段を具備することを特徴とする不良解析装置。

【請求項6】 請求項5の不良解析装置において、不良の個数が1であるか否かを判断する判断手段と、その判断手段が1であると判断するとその不良が1ビット不良であるかどうかを判断する1ビット不良判定手段とを具備することを特徴とする不良解析装置。

【請求項7】 請求項1～6のいずれかに記載の不良解析装置において、
上記二次元ウェーブレット変換手段の基底関数はH a a

r Wavelet変換のシフトパラメータを1にした冗長Haar Wavelet変換であることを特徴とする、不良解析装置。

【請求項8】 入力フェイルビットマップを二次元ウェーブレット変換し、
その変換結果ハイパス情報を含む成分について、そのハイパス情報生成処理方向にウェーブレット係数を積算してヒストグラムを生成し、
そのヒストグラムを観察することにより、フェイル・アドレスやフェイル数を解析することを特徴とする不良解析方法。

【請求項9】 請求項8記載の方法において、前記ヒストグラム中の積算値が0でないアドレスにもとづきフェイル・アドレスを求め、そのアドレスと対応するアドレスのヒストグラム積算値からフェイルビットのフェイル数を求めることを特徴とする不良解析方法。

【請求項10】 請求項9記載の方法において、上記求めたフェイル・アドレスのフェイルビットがその数からフェイルビットマップの対応アドレスの全体に渡っているか否かを判断し、

全体に渡っていない場合は、そのフェイル・アドレスにおける上記入力フェイルビットマップ又はその上記ウェーブレット変換結果を走査して不良が始まるアドレスと不良が終わるアドレスとそれら間のフェイル数を求めることを特徴とする不良解析方法。

【請求項11】 請求項10記載の方法において、フェイルビットのフェイル数が1であった場合は、上記ハイパス情報生成処理方向と直角方向におけるハイパス情報生成処理方向にウェーブレット係数を積算して第2ヒストグラムを生成し、その第2ヒストグラムについて、同様の処理を行って、そのフェイルアドレスとフェイル数を求め、これらと先に求めたフェイルアドレス及びフェイル数とを用いて1ビットフェイルであるかどうかを判断することを特徴とする不良解析方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 この発明は、例えば集積回路の検査工程などで作成されるフェイルビットマップから、不良の特徴を検出する、不良解析方法および装置に関する。

【0002】

【従来の技術】 集積回路の検査工程において、DRAMなどのように“メモリがアレイ状に配列している集積回路”の検査結果は、正常に動作している部分であることを意味するパスビットと、正常に動作していない部分、すなわち不良を意味するフェイルビットから構成される。検査結果は半導体検査装置から、フェイルビットマップとして出力される。このフェイルビットマップを利用して、フェイル・アドレスやフェイル数を特定し、発生原因を解析して製造ラインへフィードバックすること

は、半導体製品の歩留まりを向上させる上で非常に重要である。通常、このフェイルビットマップの解析は、例えばパスビットが0、フェイルビットが1というような二値画像を作成して可視化することにより実現する。可視化した情報を人間が観察し、どこにどのような不良原因があるのかを、解析する。しかし、回路の高集積化が進むにつれて、フェイルビットマップは膨大になる。例えば1024×1024ビットのフェイルビットマップになると、512×512画素の解像度をもつ表示装置では1画面で表示することができない。このため、1枚のフェイルビットマップを数画面に分けて表示する必要がある。サイズが膨大な画像の、任意のアドレスに発生しうる欠陥を、人間が細かく観察する必要があるので、解析作業が繁雑となる。また、フェイル・アドレスやフェイル数の測定に誤りが生じるおそれがあった。このため、メモリデバイスの容量が1Mbit以上の場合のように、表示データが大きいときのフェイルビットマップを1画面に表示して、解析作業を容易にする技術、あるいは、従来人間がおこなっている、フェイルビットマップの解析作業そのものを自動化するような技術の必要性が高まってきている。

【0003】従来、表示データが大きいフェイルビットマップを1画面に表示するために、フェイルビットマップの二値画像を圧縮して表示する技術が利用されている。これは、フェイルビットマップを例えば1ブロックが2×2ビットのブロックに分割して、1ブロックを1画素で表示する。各画素は、対応するブロックに含まれるフェイルビットの数によって、色分けして表示する。例えば1ブロックが2×2ビットであれば、1ブロック内のフェイルビットの数は0～4であるから、5色をつかって圧縮したフェイルビットマップを表示する。これにより、512×512画素の表示装置に1024×1024ビットのフェイルビットマップを表示して、データ量が大きいフェイルビットマップを1画面で解析することが可能となる。さらに、集積度が増して、2048×2048ビットのフェイルビットマップを解析する場合は、1ブロックを4×4ビットとして圧縮することにより、1画面で解析することが可能となる。

【0004】

【発明が解決しようとする課題】フェイルビットマップを圧縮して表示する技術を利用した場合の問題点を説明

$$\begin{aligned} X(0) &= g_0 \cdot x(0) + g_1 \cdot x(1), \\ X(1) &= g_0 \cdot x(2) + g_1 \cdot x(3), \\ X(2) &= g_0 \cdot x(4) + g_1 \cdot x(5), \\ &\vdots \\ &\vdots \\ &\vdots \end{aligned}$$

$$X(7) = g_0 \cdot x(14) + g_1 \cdot x(15), \quad X(15) = h_0 \cdot x(14) + h_1 \cdot x(15)$$

このHaar Wavelet変換に具体的な信号を入力してみる。たとえば、つぎの入力信号 $x(n)$ をあた

する。たとえば、2048×2048ビットのフェイルビットマップを512×512画素に圧縮する場合は、1画素に4×4ビットの情報をもたせる必要がある。各画素はフェイル数によって色分けするので、この場合は17色に色分けする必要がある。今後、メモリデバイスの容量が256Mbit、1Gbitとますます増大するにつれて、1画素を表示するのに必要な色数がメモリデバイスの容量に比例して増加する。1画素の色数が増え、画素の色からフェイルビットの数を把握することが困難になってくるという問題がある。また、従来の技術は、圧縮したフェイルビットマップを画面上に表示し、最終的に人間が観察するので、フェイル・アドレスや、フェイル数を人間の目で細かく測定しなければいけないという、繁雑さは解決されない。

【0005】この発明の目的は、従来、人間がおこなっている、フェイル・アドレスやフェイル数の解析作業を自動的におこなう不良解析方法および装置を提供することにある。この発明の説明に先立ちウェーブレット変換について説明する。

20 A. ウェーブレット変換手段

ウェーブレット変換についてはCHUI, "An Introduction to WAVELETS," Academic Press, 1992に詳しく記載、説明されているが、以下に簡単に説明する。まず、一次元ウェーブレット変換について説明する。ウェーブレット変換をおこなう際に使用する基底関数は多数存在するが、ここではその中で基底関数のフィルタ長がもっとも短いHaar Wavelet (ハール ウェーブレット)を用いて説明する。他のウェーブレット基底関数についても関数の形が違っただけで、出力される情報はほぼ同様になる。ウェーブレット変換はスケーリング関数とウェーブレット関数の二つの直交する関数により構成される。スケーリング関数はデータの平滑情報 (=ローパス情報)を出力する関数で、ウェーブレット関数はデータの詳細情報 (=ハイパス情報)を出力する関数である。Haar Waveletの場合は、スケーリング関数が $g_0 = g_1 = 1/2$ で、ウェーブレット関数が $h_0 = 1/2$ 、 $h_1 = -1/2$ である。

【0006】入力信号 $x(n)$ ($0 \leq n \leq 15$; n は整数)のHaar Wavelet変換を適用した結果 $X(N)$ はつぎのようになる。

$$\begin{aligned} X(8) &= h_0 \cdot x(0) + h_1 \cdot x(1) \\ X(9) &= h_0 \cdot x(2) + h_1 \cdot x(3) \\ X(10) &= h_0 \cdot x(4) + h_1 \cdot x(5) \end{aligned}$$

50 【0007】

5

6

$$x(n) = \{0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0\} \quad (1)$$

↑

フェイルビット

この信号は1か所だけで値が1となっており、残りは0である。この明細書では、値が1の部分フェイルビットと呼ぶ。また、フェイルビットの位置のことをフェイルビット位置と呼ぶ。

ル・アドレスと呼ぶ。

【0008】(1)の信号をHaar Wavelet変換すると、つぎの結果X(N)をえる：

$$X(N) = \{0, 0, 0, 1/2, 0, 0, 0, 0 \mid 0, 0, 0, -1/2, 0, 0, 0, 0\} \quad (2)$$

スケーリング関数 ウェーブレット関数
(ローパス成分) (ハイパス成分)

ウェーブレット変換した結果をウェーブレット係数とよぶ。ハイパス成分のウェーブレット係数X(11)に入力信号x(n)のフェイルビットが検出されている。このように、ウェーブレット変換には、フェイルビットを検出する性質がある。

【0009】つぎに、(1)であたえた入力信号のフェイル・アドレスを1個右にシフトした(3)をあてて、これにHaar Wavelet変換を適用すると、(4)の結果をえる。

$$x(n) = \{0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0\} \quad (3)$$

↑

フェイルビット

$$X(N) = \{0, 0, 0, 0, 1/2, 0, 0, 0 \mid 0, 0, 0, 0, 1/2, 0, 0, 0\} \quad (4)$$

スケーリング関数 ウェーブレット関数
(ローパス成分) (ハイパス成分)

このように1の位置が1サンプルずれると、ハイパス成分のウェーブレット係数の符号が反転する。

【0010】入力信号(1)、(3)はデルタ関数となっている。一般にデルタ関数 $\delta(\tau - k)$ { $k = 2i -$

1, $k = 2i$; i は自然数}のHaar Wavelet変換のハイパス成分X(N)は(5)式であたえられる。

$$\begin{aligned} k = 2i - 1 \text{ で } X(N) &= (1/2) \delta(\tau - (k-1)/2) \\ k = 2i \text{ で } X(N) &= (-1/2) \delta(\tau - k/2) \end{aligned} \quad (5)$$

このように、フェイル・アドレスによって、ウェーブレット係数の符号が変化する。逆に、ハイパス成分にフェ

イルビット(S/2) $\delta(\tau - j)$ のアドレスjと、その符号Sから、入力信号のフェイル・アドレスkは、

$$\begin{aligned} S > 0 \text{ の場合, } k &= 2j, \\ S < 0 \text{ の場合, } k &= 2j + 1 \end{aligned} \quad (6)$$

と求めることができる。

B. 冗長Haar Wavelet変換

以上に示したように、ウェーブレット変換を利用して、フェイル・アドレスを検出することができるが、Haar Wavelet変換を使った場合、フェイル・アドレスを正確に求めるには、ウェーブレット係数の符号によって場合分けする必要がある。つぎに示す冗長Haar Wavelet変換を利用すると、場合分けすることなく、フェイル・アドレスを正確に求めることができる。

【0011】通常のHaar Wavelet変換は入力信号にたいして基底関数を信号2個分シフトしてサンプリングをおこなうが、冗長Haar Wavelet変換は、ハイパス成分の演算において、入力信号にたいして基底関数を信号1個分だけシフトすることを繰り返してサンプリングをおこなうことにより実現する。入力信号x(n)を冗長Haar Wavelet変換すると、変換結果X(N)をえる。

40 【0012】

ローパス成分 (スケーリング関数)	ハイパス成分 (ウェーブレット関数)
$X(0) = g_0 \cdot x(0) + g_1 \cdot x(1),$	$X(8) = h_0 \cdot x(0) + h_1 \cdot x(1)$
$X(1) = g_0 \cdot x(2) + g_1 \cdot x(3),$	$X(9) = h_0 \cdot x(1) + h_1 \cdot x(2)$
$X(2) = g_0 \cdot x(4) + g_1 \cdot x(5),$	$X(10) = h_0 \cdot x(2) + h_1 \cdot x(3)$
.	.
.	.
.	.
$X(7) = g_0 \cdot x(14) + g_1 \cdot x(15),$	$X(23) = h_0 \cdot x(14) + h_1 \cdot x(15)$

この冗長Haar Wavelet変換を入力信号 δ
($\tau-k$)に適用すると、そのハイパス成分 $X(N)$ は

$$X(N) = \{(-1/2)\delta(\tau-k) + (1/2)\delta(\tau-k-1)\}$$

このように、1つのフェイルビットから、正と負の2つのウェーブレット係数が検出されるので、どちらかの符号のウェーブレット係数だけを抽出することにより、フェイル・アドレスを正確に求めることが可能となる。つまり(7)式で δ 関数が求まるアドレス k 自体がフェイル・アドレスとなる。

C. 二次元ウェーブレット変換

ウェーブレット変換は、2次元の画像データに対しても適用することができる。具体的な画像データにウェーブレット変換を適用した例を、前記文献その他ウェーブレット変換を説明したものに示されている。

【0014】図8(a)に示す原画像51が例えば512×512のデジタルデータとなっているとする。この画像に対して、まずX軸方向に対して一次元ウェーブレット変換をほどこす、つまりX軸方向の512個の信号に対するウェーブレット変換を、Y軸方向に512回繰り返す。これにより図8(b)の画像が得られる。図8(b)には、原画像図8(a)の原画面を縦(左、右)に2分割し、左側にローパス情報(L)52L、右側にハイパス情報(H)52Hが格納されている。この図8(b)の画像に対して今度はY軸方向に同じウェーブレット変換をほどこす。これにより図7(c)の画像が得られる。図8(c)には、図8(b)の画面を横(上、下)に2分割し、上側にローパス情報(L)、下側にハイパス情報(H)が格納されている。結果として図8(c)には、原画面を4分割し、左上にX、Y軸両方向のローパス情報(LL成分)53LLが、右上にX軸方向のハイパスとY軸方向のローパスを組み合わせたもの(HL成分)53HLが、左下にX軸方向のローパスとY軸方向のハイパスを組み合わせたもの(LH成分)53LHが、右下にはX、Y軸両方向のハイパス情報(HH成分)53HHがそれぞれ格納されている。

【0015】ハイパスフィルタは、画素値の局所的な変化分を算出する。LH成分53LHは垂直方向にハイパスフィルタを適用するので、垂直方向の画素値の変化、すなわち横方向に走るエッジ成分を検出している。つまり、2次元ウェーブレット変換では、LH成分53LHが横方向のエッジ(以降横線成分と呼ぶ)を検出し、HL成分53HLが縦方向のエッジ(以降縦線成分と呼ぶ)を検出する。また、X、Y両方向にハイパスフィルタを適用したHH成分53HHには、45°方向のエッジ(斜線成分と呼ぶ)が検出される。この明細書では、まずX軸方向に変換処理をおこなってからY軸方向に変換処理をおこなって図8(c)の処理画像を得ているが、変換の順序を入れ替えても、二次元ウェーブレット変換は全く同じ図8(c)の画像を得る。

(7)式であたえられる。

【0013】

(7)

【0016】この二次元ハールウェーブレット変換の処理手順の例を図9に示す。二次元画像 $V(i, j)$ ($i=1, 2, \dots, M, j=1, 2, \dots, N, i$ はx軸アドレス、 j はy軸アドレスと対応)が入力され(S1)、まずレベル $q, q=\log_2 M, \log_2 M-1, \dots, 2, 1$ が計算され(S2)、その各レベル q についてローパス演算(S3)とハイパス演算(S4)とが行われる。つまりローパス演算では $X_L(i, y) = \{V(2i-1, y) + V(2i, y)\}/2$ ($i=1, 2, \dots, 2^{q-1}$)が演算される。最初のX方向(水平方向)の変換演算は i が1から 2^{q-1} 、つまりこの $\log_2 M$ 乗、つまり M までのローパス演算 $X_L(i, y)$ とハイパス演算 $X_N(i, y)$ とが行われる。

【0017】次に $q_y = q + \log_2(N/M)$ が演算され(S5)、 $X_L(i, y)$ についてローパス演算 $X_{LL}(i, j) = \{X_L(i, 2j-1) + X_L(i, 2j)\}/2$ ($j=1, 2, \dots, 2^{q_y-1}$)が演算され(S6)、またハイパス演算 $X_{LH}(i, j) = \{-X_L(i, 2j-1) + X_L(i, 2j)\}/2$ ($j=1, 2, \dots, 2^{q_y-1}$)とが演算される(S7)。 $q=\log_2 M$ の時は、これら演算 $X_{LL}(i, j), X_{LH}(i, j)$ は $j=1$ から N まで行われることになる。同様に $X_H(i, y)$ に対してローパス演算 $X_{HL}(i, j) = \{X_H(i, 2j-1) + X_H(i, 2j)\}/2$ ($j=1, 2, \dots, 2^{q_y-1}$)がなされ(S8)、またハイパス演算 $X_{HN}(i, j) = \{-X_H(i, 2j-1) + X_H(i, 2j)\}/2$ ($j=1, 2, \dots, 2^{q_y-1}$)がなされる(S9)。この場合も $q=\log_2 M$ の時は、 $j=1$ から N まで行われる。

【0018】

【課題を解決するための手段】この発明は、入力フェイルビットマップの二値画像に対して二次元ウェーブレット変換を適用、その変換結果を画像として観察し、あるいはこのウェーブレット変換結果から、フェイル・アドレスやフェイル数といったフェイルビット情報を自動的に抽出して出力する。

【0019】フェイルビットマップの二次元ウェーブレット変換によればフェイルビット情報が検出できることを具体的に説明する。図1(a)にフェイルビットマップの一例を示す。フェイルビットは図示のように水平方向に走る直線41や垂直方向に走る直線42、あるいは孤立点43として存在している。このフェイルビットマップに二次元ウェーブレット変換を適用すると、図1(b)に示すように、水平方向のフェイルビット(直線41)は右上の画像53HL中に横線成分44として検

10

20

30

40

50

出され、垂直方向のフェイルビット（直線42）は左下の画像53LH中の縦線成分45としてそれぞれ検出される。したがって、ウェーブレット変換の結果を表示・観察することにより、フェイル・アドレスやフェイル数を解析することが可能になる。また各成分は、それぞれ入力画像の1/4のサイズになっているので、メモリデバイスの容量が大きくなっても、フェイルビット情報を1画面に表示して解析することが可能となる。

【0020】従って、この発明によれば図2に示すように入力フェイルビットマップを二次元ウェーブレット変換し（S1）、その二次元ウェーブレット変換結果中のX方向ハイパス、Y方向ローパス情報 X_{HL} に対し、その各Xアドレスについてウェーブレット係数を積算してXアドレスヒストグラムを生成し、またX方向ローパス、Y方向ハイパス情報 X_{LH} に対し、その各Yアドレスについてウェーブレット係数を積算してYアドレスヒストグラムを生成する（S2）。そのXアドレスヒストグラム、YアドレスヒストグラムのそれぞれにおけるXアドレス、Yアドレスで積算値が0でないものから、不良Xアドレス、不良Yアドレスをそれぞれ検出し（S3）、更にその各不良アドレスにおける積算値からフェイル数を求める（S4）。

【0021】

【発明の実施の形態】この発明の実施例を図3に示す。入力されたフェイルビットマップを二次元ウェーブレット変換する。説明を簡単にするためY方向フェイルビットの情報、つまり、縦線成分を含むX方向ハイパス、Y方向ローパス情報 $X_{HL}(i, j)$ を取り込む（S1）、その画像53HLは例えば図3（b）に示すように、垂直（Y）方向ラインフェイルが、縦線45として現われる。

【0022】図には示していないが水平方向のフェイルビットの情報検出も同様におこなわれる。次に、ウェーブレット変換縦線成分のウェーブレット係数を縦方向に積算して、ヒストグラムを作成する（S2）。ヒストグラム生成手段の処理フローを図4（a）に示す。つまり図9中の垂直方向変換におけるローパス情報 X_{HL} について、 $H_{HL}(i) = \sum X_{HL}(i, j)$ （ \sum は $j=1$ から 2^{qy-1} まで）を演算する。つまり各Xアドレス i について、Yアドレス $j=1$ から 2^{qy-1} までそれぞれにおけるウェーブレット係数 $X_{HL}(i, j)$ を積算する。前記例のように $M=N=512$ の場合、 $i=1$ から $i=256$ の各 i について、 $j=1$ から $j=256$ の各 j との (i, j) における $X_{HL}(i, j)$ を積算することになる。このようにしてXアドレスヒストグラムが生成される。ウェーブレット変換画像53HLに対するXアドレスヒストグラムは図3（c）に示すようになる。図4（b）に垂直方向変換におけるハイパス情報 X_{LH} に対するYアドレスヒストグラムの生成処理を示す。

【0023】作成されるXアドレスヒストグラムは、垂

直（Y）方向のフェイルビットが存在するアドレスで、0でない値となり、フェイルビットが存在しないアドレスでは0となる。そこで、得られたヒストグラムを走査して（S3）、積算値が0でないアドレスを探索する

（S4）。0以外の積算値が検出された場合は、このアドレスを、垂直方向フェイルビットのXアドレスとしてメモリに保存する。Haar Wavelet変換を利用している場合は、A. 節で述べたように、フェイルビットのXアドレスが偶数か奇数かによって、積算値の符号が変化するので、（6）式をもちいて、正確なXアドレスをもとめることができる。冗長Haar Wavelet変換を利用する場合は、0以外の積算値の符号が正になる部分だけ、あるいは負になる部分だけに着目する。冗長Haar Waveletを利用して検出されたアドレスを $Xaddr$ としたとき、実際のフェイルビットのXアドレスは、

正の積算値に着目したとき： $Xaddr-1$

負の積算値に着目したとき： $Xaddr$

であるとして、メモリに保存する。

【0024】つぎに、フェイル数を算出する（S5）。フェイル数は、ヒストグラムの積算値で表されている。Haar Wavelet変換を利用する場合は、積算値の絶対値を2倍した値、冗長Haar Wavelet変換を利用する場合は、積算値の絶対値自体が、フェイル数となる。しかし、求められたフェイル数がフェイルビットマップのサイズより小さい場合、それが1本の連続したフェイルビットであるのか、複数本のフェイルビットの集まりであるかの判断と、各フェイルビットのYアドレスを求める必要がある。これを実現する処理を次に示す。

【0025】まず、算出されたフェイル数が入力フェイルビットマップのサイズに等しい場合（S6）は、フェイルビットマップの一番上から一番下まで通っている一本の線状のフェイル、すなわちラインフェイルであると考えられるので、ラインフェイルのサイズとそのXアドレスをフェイルビット情報として出力する（S7）。算出されたフェイル数が入力フェイルビットマップのサイズより小さい場合は、図5に示すように以下の処理をおこなう。

a) フェイルビットマップのウェーブレット変換縦線成分、または入力フェイルビットマップ中の、当該ラインフェイルのXアドレスにおいて、上から下に、あるいは下から上にウェーブレット係数またはビットマップを走査する（S8）。

b) その走査によりウェーブレット係数またはビット値がパスビットからフェイルビットへの変化を検出すると、その変化したYアドレスを、ラインフェイルの始まりとして、メモリに登録する（S9）。

c) ウェーブレット係数またはビット値がフェイルビットである区間の大きさをカウントしながら、走査を続け

る(S10)。

d) ウェーブレット係数またはビット値がフェイルビットからパスビットへの変化を検出すると、そのYアドレスを、ラインフェイルの終わりとして、メモリに登録する(S11)。

e) カウントされたラインフェイルのフェイル数の合計が、当該Xアドレス全体でのラインフェイルのフェイル数より小さければb)～d)(S9～S13)の処理を繰り返し、等しければ処理を終了する(S13)。

【0026】以上の処理により、ラインフェイルが1本であっても複数本であっても、それぞれのラインフェイルのアドレスとフェイル数を自動的に求めて出力することが可能となる。特殊な場合として、フェイル数が1のフェイルビットが検出される場合がある(S12)。垂直方向にフェイル数1のフェイルビットが検出される要因には以下の2通りがある。

【0027】α) 孤立している1ビットフェイル

β) 水平方向のラインフェイルの始点または終点

これを判別するために、水平方向のラインフェイルの検出結果を利用する。即ち図6に示すように、ウェーブレット変換の横線成分 $X_{LH}(i, j)$ にたいして、図3に示したと同様の処理を適用し(S1)その結果から水平方向1ビットフェイル候補を求め(S2)先の処理結果から垂直方向1ビットフェイル候補を求め(S3)。

α) の場合は、縦線成分への処理で求められた1ビットフェイルと全く同じアドレスに1ビットフェイルが検出されるが、β) の場合はラインフェイルの始まりまたは終わりのアドレスとして検出される。したがって、縦線成分、横線成分の両方から、同じアドレスに1ビットフェイル候補が検出されたときのみ(S4)、1ビットフェイルとして、そのアドレスを出力し(S5)、1ビットフェイルがどちらかにしか検出されなかった場合は、無視する。

【0028】以上説明したように、二次元ウェーブレット変換を利用することにより、フェイルビットマップから、フェイル・アドレスやフェイル数といった情報を自動的に検出することが可能となる。図7はこの発明にもとづく装置の機能的構成を示す。コンピュータ101は、入出力や計算をおこなうCPU31と、よく使う命令やデータを保持して処理の高速化をおこなうキャッシュ32、浮動小数点演算部33、ユーザ入力やデータを記憶するRAM34やROM36、ユーザの選択メニューや計算結果を表示する表示装置35、パラメータや命

垂直方向のラインフェイル: Xアドレス31, 127の2本

水平方向のラインフェイル: Yアドレス63の1本

残りの5種類のフェイルビットについて、フェイルビットの始まりと終わりを求めると次のようになったとす

縦線成分:

	始まりのYアドレス	終わりのYアドレス	(Xアドレス)
(i)	31	32	(9)

令を入力するキーボードや、ポインティングデバイス(マウスなど)の入力装置37、計算結果などを保存する磁気ディスク38を備えている。

【0029】検査対象となるフェイルビットマップは、半導体検査装置41により生成され、又はフェイルビットマップデータからバスを経由してコンピュータ上のRAM34に転送される。入力フェイルビットマップの二値画像は例えば図1(a)に示すようなものでサイズは512×512の場合である。RAM34に格納されたフェイルビットマップを入力データとして、まずこれをフレームメモリ61に転送する。転送したこのフェイルビットマップをウェーブレット変換手段51により二次元ウェーブレット変換をおこない、変換結果をフレームメモリ62に転送する。つぎにヒストグラム生成手段52にて水平方向、垂直方向のヒストグラムを生成し、コンピュータ上のRAM34に転送する。つぎにこのヒストグラムを調べて、積算値が0でない部分のアドレスと積算値を算出する。図3(b)に示したHaar Wavelet変換結果の縦線成分に対しヒストグラム生成したXアドレスヒストグラムが図3(c)に示したものとなり、このヒストグラムから、以下の情報が得られ、

	Xアドレス	積算値
(i)	4	-0.5
(ii)	8	11.5
(iii)	15	-256
(iv)	63	-256

同様に、横線成分のYアドレスヒストグラムからは以下の情報が得られたとする。

【0030】

	Yアドレス	積算値
(v)	0	-0.5
(vi)	7	5
(vii)	12	0.5
(viii)	31	-256

積算値に着目すると、-256であるラインフェイルが(iii), (iv), (viii)の3本存在する。その絶対値の2倍が512になるので、これはフェイルビットマップの端から端に通っていると判断する。これと、積算値の符号から、(6)式をつかって、次の3本のラインフェイルのアドレスが検出される。

【0031】

13

14

(ii)	1	2 4	(1 6)
横線成分：			
	始まりのXアドレス	終わりのXアドレス	(Yアドレス)
(v)	8	9	(1)
(vi)	9	2 0	(1 4)
(vii)	8	9	(2 3)

(i), (v), (vii)については、フェイルビットのフェイル数が1である。しかし、縦線成分、横線成分で、X、Yアドレスが共に等しい1ビットフェイルは存在していないので、これらは無視する。

【0032】この結果、先に求めた3本のラインフェイルと、次に示す2本のラインフェイルの計5本のラインフェイルが入力フェイルビットマップに存在していることが最終的に検出される。

垂直方向のラインフェイル：Xアドレス16、始まりのYアドレス1、終わりのYアドレス24、フェイル数23

水平方向のラインフェイル：Yアドレス14、始まりのYアドレス9、終わりのYアドレス20、フェイル数21

以上の情報を検出結果として、コンピュータの表示装置35に出力する。

【0033】図7では、ウェーブレット変換手段51やヒストグラム生成手段52をコンピュータ101から独立した装置として構成したが、これらのウェーブレット変換、ヒストグラム生成処理もコンピュータ101のCPU31による処理で代替させてもよい。

【0034】

【発明の効果】請求項1及び2の発明はこの発明の第1の目的に対応するものであり、従来の画像圧縮手法では失われてしまう、フェイル・アドレス情報を、この発明では保存したまま、フェイルビット解析をおこなうことが可能である。請求項3乃至8の発明はこの発明の第2の目的に対応するものであり、従来人間がおこなっているフェイル・アドレスやフェイル数の解析作業を、この発明では自動的におこなうことが可能である。

【0035】以上詳細に説明したように、この発明は半

導体試験装置又はフェイルビットマップデータベースより出力されるフェイルビットマップに存在する、フェイル・アドレスやフェイル数といった情報を、二次元ウェーブレット変換を利用することにより、自動的に検出することが可能となり、検査工程自動化の面からその効果は大である。

【図面の簡単な説明】

【図1】(a)はフェイルビットマップのサンプル画像を示す図、(b)は図1(a)の画像を二次元ウェーブレット変換した結果の画像を示す図である。

【図2】この発明の方法の基本処理を示す流れ図。

【図3】(a)はこの発明方法の実施例を示す流れ図、(b)はウェーブレット変換した結果中のX方向ハイパスY方向ローパス情報(縦線検出成分)の画像例を示す図、(c)はそのヒストグラム例を示す図である。

【図4】ヒストグラム生成処理手順を示す図。

【図5】図4中のステップS4、S5と対応し、ラインフェイルのアドレスとフェイル数を求める処理手順の例を示す流れ図。

【図6】ラインフェイルのフェイル数が1だった場合に、そのフェイルビットが1ビットフェイルであるかどうかを判別する処理手順の例を示す流れ図。

【図7】この発明にもとづく画像情報処理装置の機能構成例を示す図。

【図8】原画像とその二次元ウェーブレット変換画像と、二次元ウェーブレット変換画像との関係を説明するための図。

【図9】二次元画像V(x, y)にたいする二次元Haar Wavelet変換手段の処理手順を示す流れ図。

10

20

30

【図1】

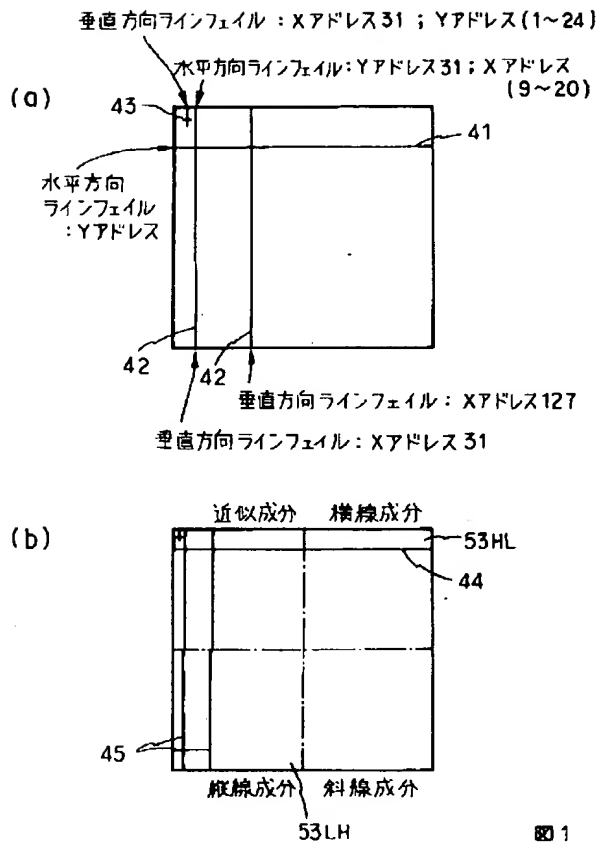


図1

【図6】

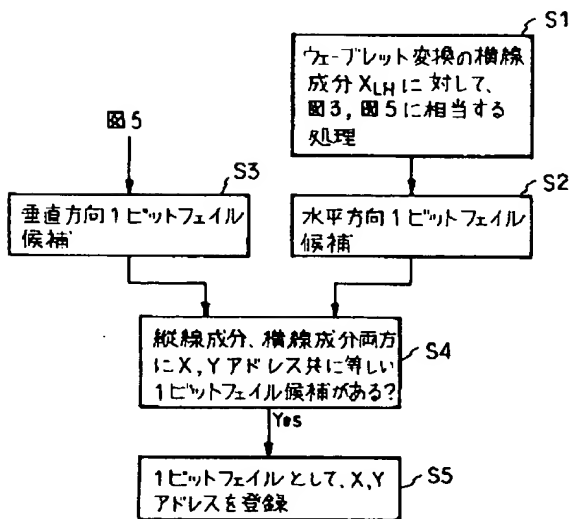


図6

【図2】

入力ファイルビットマップ

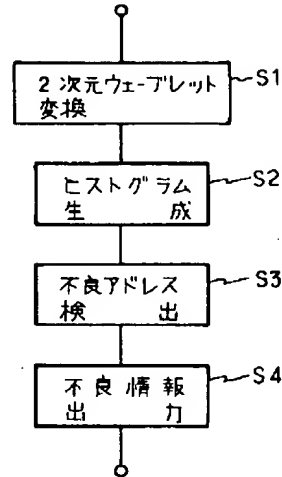


図2

【図3】

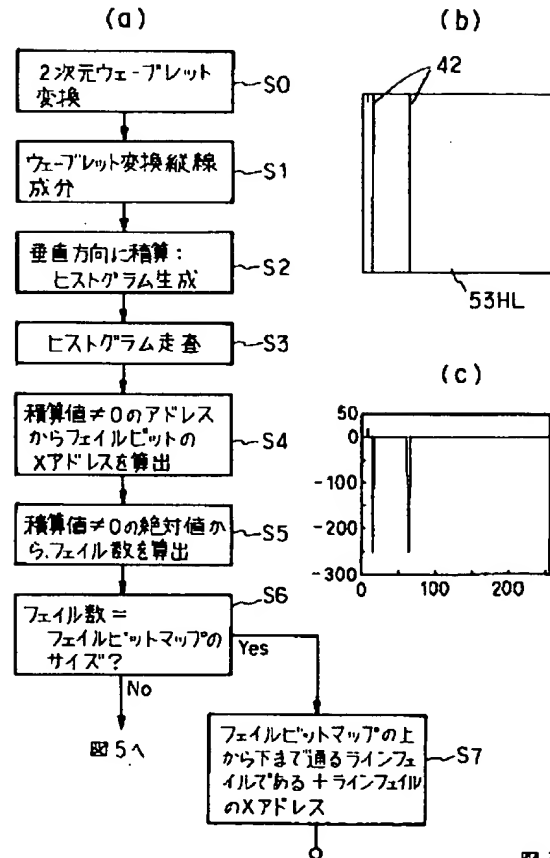


図3

【図 4】

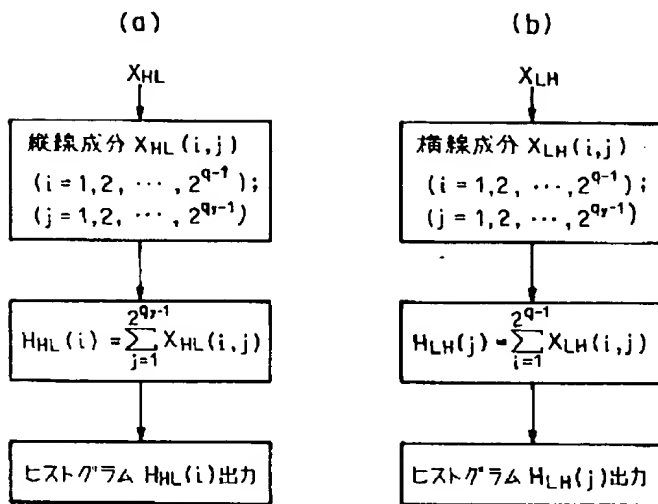


図 4

【図 5】

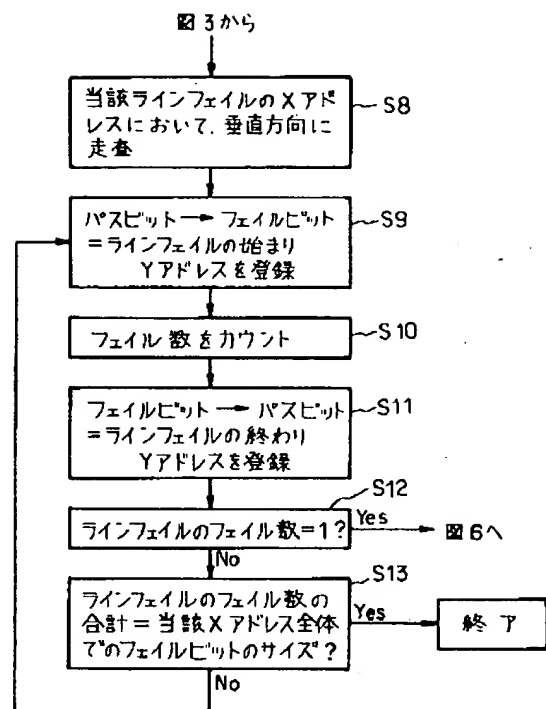


図 5

【図 7】

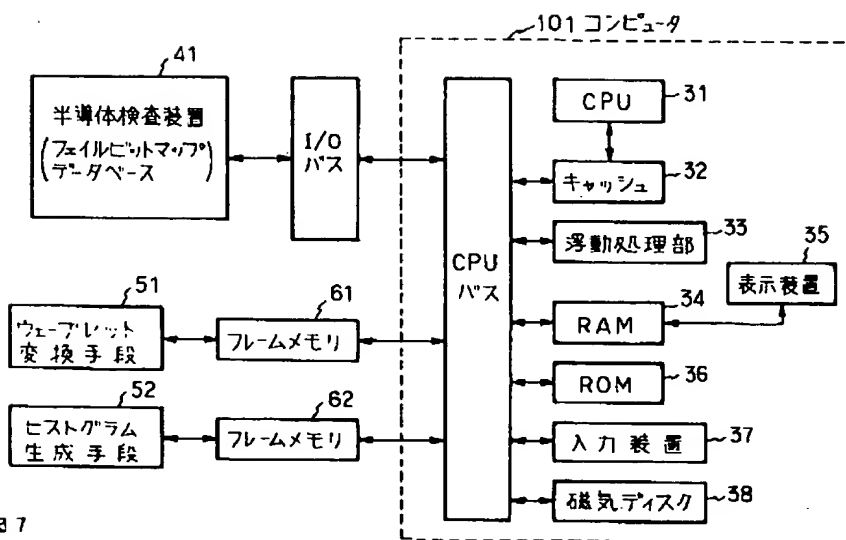


図 7

【図 8】

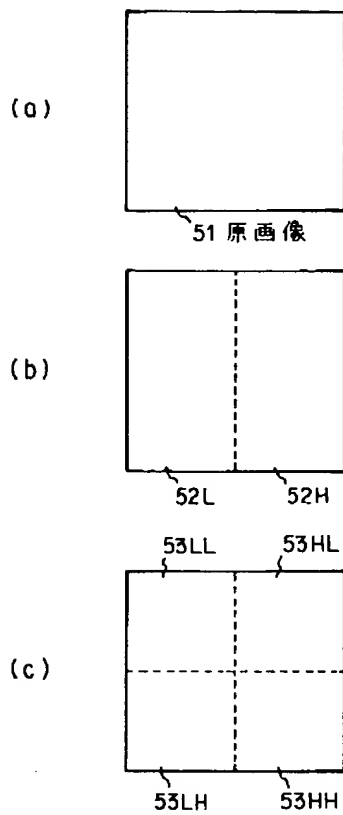


図 8

【図 9】

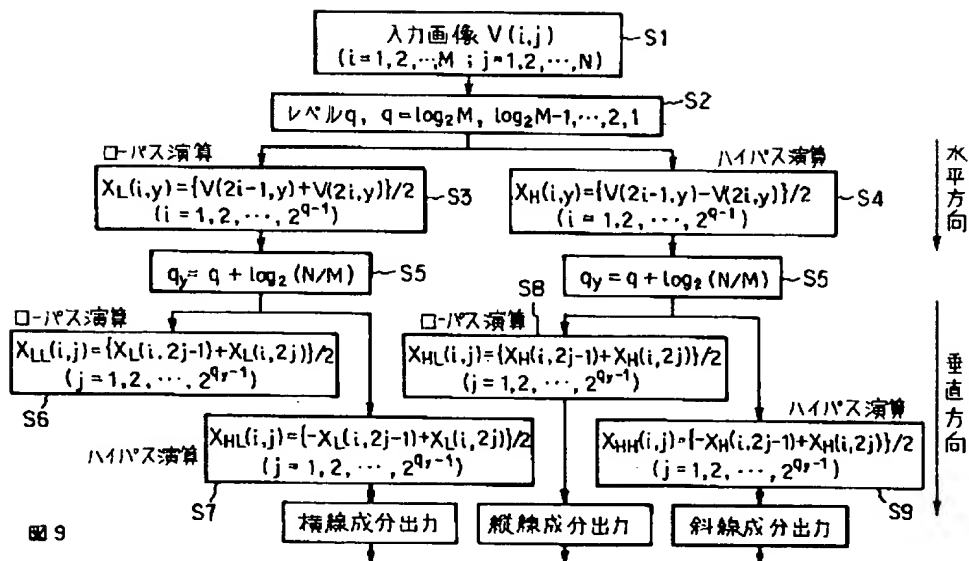


図 9